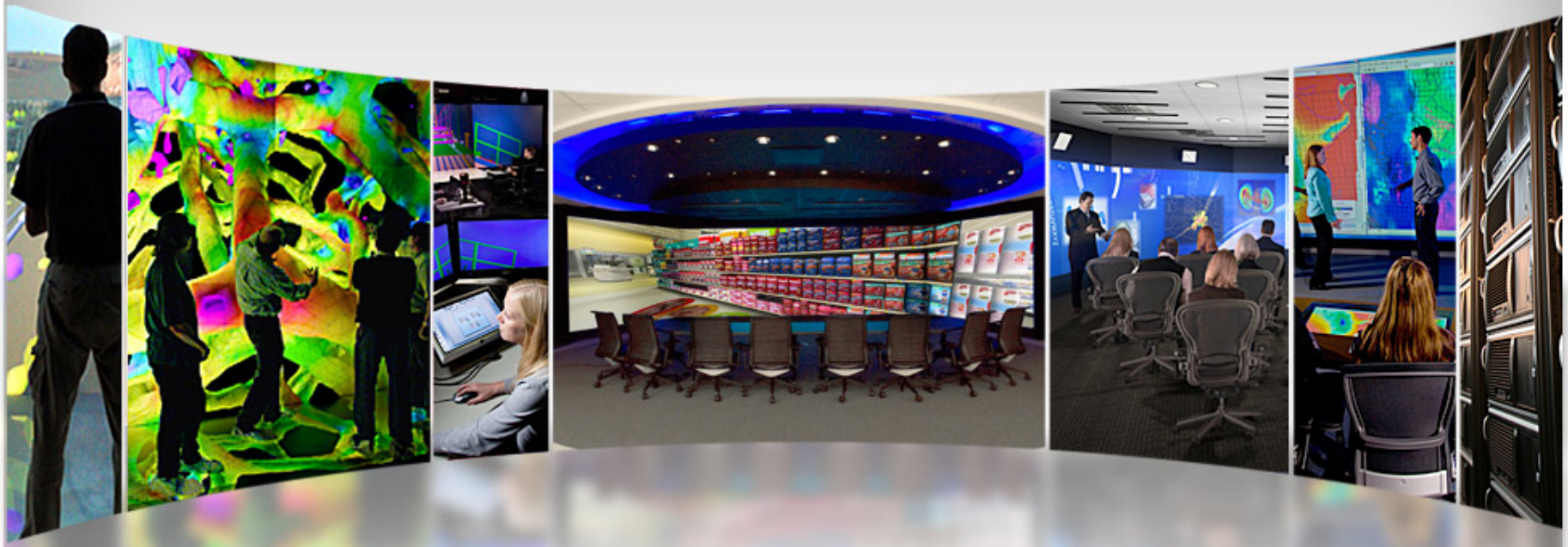


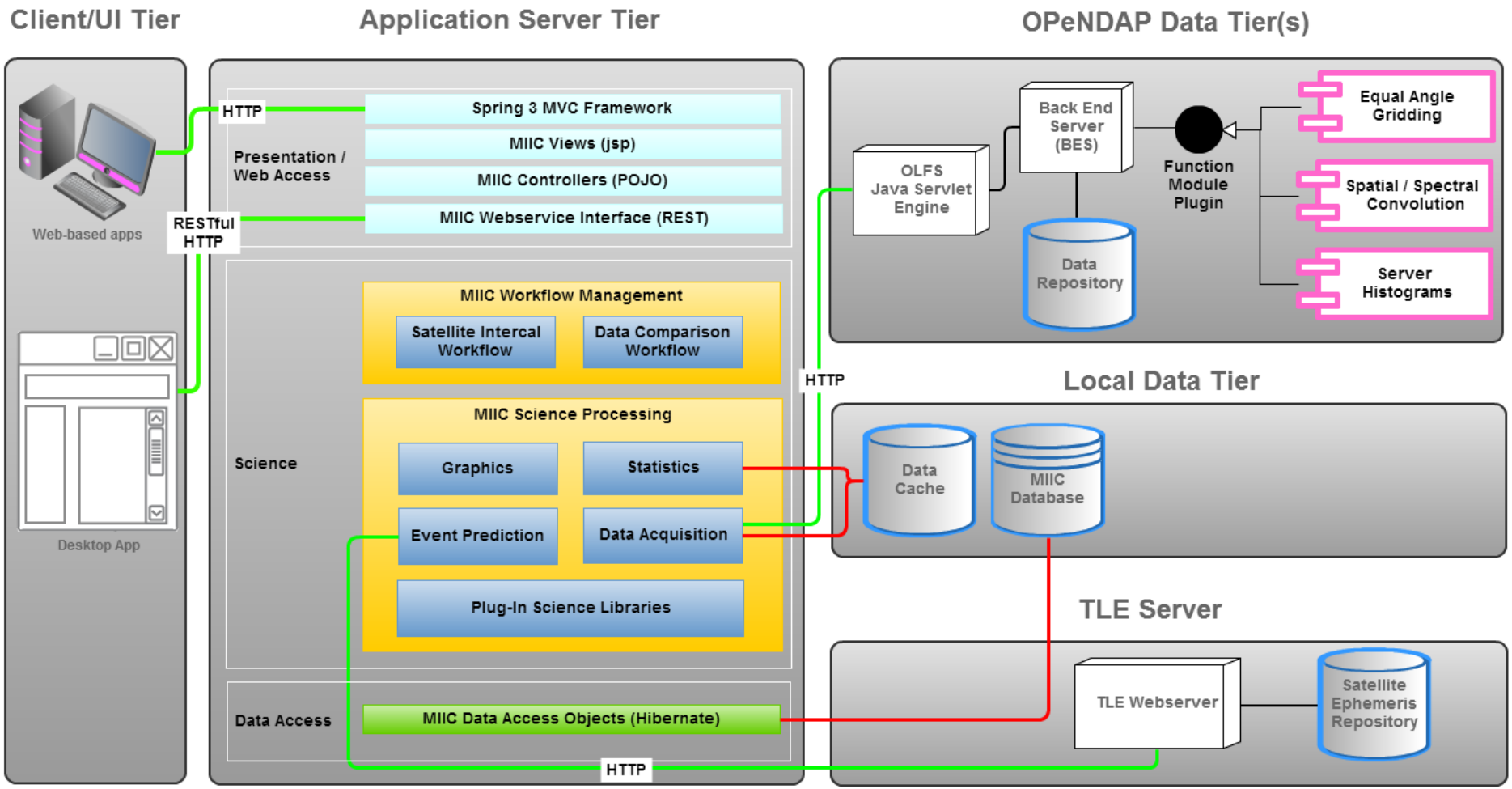
Multi-Instrument Intercomparison

Design & Deployment

Mechdyne

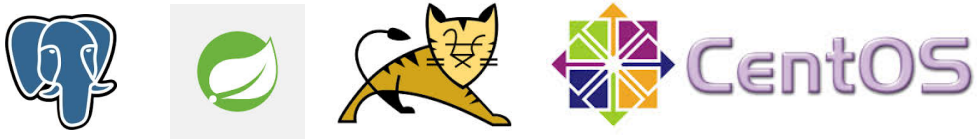
ENABLING DISCOVERY

Multi-Tier Architecture



Technology Stack: MIIC Tier

- PostgreSQL-Spring-Tomcat-Linux
- Spring Framework for Java
  - Requires only *web container* (Tomcat) to deploy
  - Works well with JavaEE (JPA2, transactions, etc.)
  - Spring Model-View-Controller – Web pages & REST interface
  - Spring Security – Authentication & authorization
  - REST template, spring expression language, other stuff…
  - Core features encourage good design
    - Dependency injection – keep classes generic & free of configuration



Technology Stack: OPeNDAP

- OPeNDAP Hyrax Server w/ MIIC plugin (C++)
  - Latest server version, available as RPM
  - Current plugin version 1.3 (also available as RPM)
- Plugin is very general & should have uses beyond MIIC
  - Works on any file type supported by OPeNDAP (HDF-EOS, etc)
  - 2D histogram: return averaged data w/ statistics
    - Choose x & y axis data variables, range, and # bins
  - Tuple: return subset data (flat or retaining original shape)
  - Filter all observations by expression
  - Generate new observations from expression
  - Decimate huge data variables (i.e. skip value)

OPeNDAP expression examples

- Apply expression to variable: convert SSF co-latitude to latitude

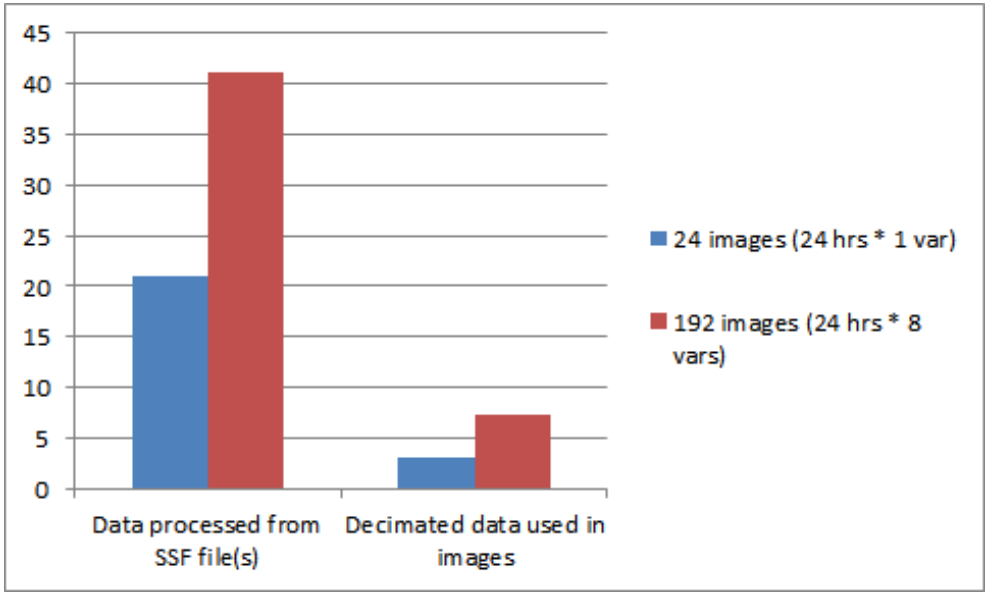
```
…
define_var(eval_expr(Time_and_Position_Colatitude_of_subsatellite_point_at_surface_at_observation,"90-val"),"latitude",0,180,UNBOUND)
```

- Filter observations using expression: SSF wavelength is not 11.03 or 12.02

```
…
define_filter_expr(Footprint_Imager_Radiance_Statistics_Imager_channel_central_wavelength,"val != 11.03 && val != 12.02")
```

OPeNDAP 2D Histogram for lat/lon charts

- From another study: volume of data (GB) to build 24 or 192 level 2 SSF images for one data var.
  - Left: Volume of file data to process
  - Right: Volume of averaged data in image
  - Process ~6x less data @ webserver



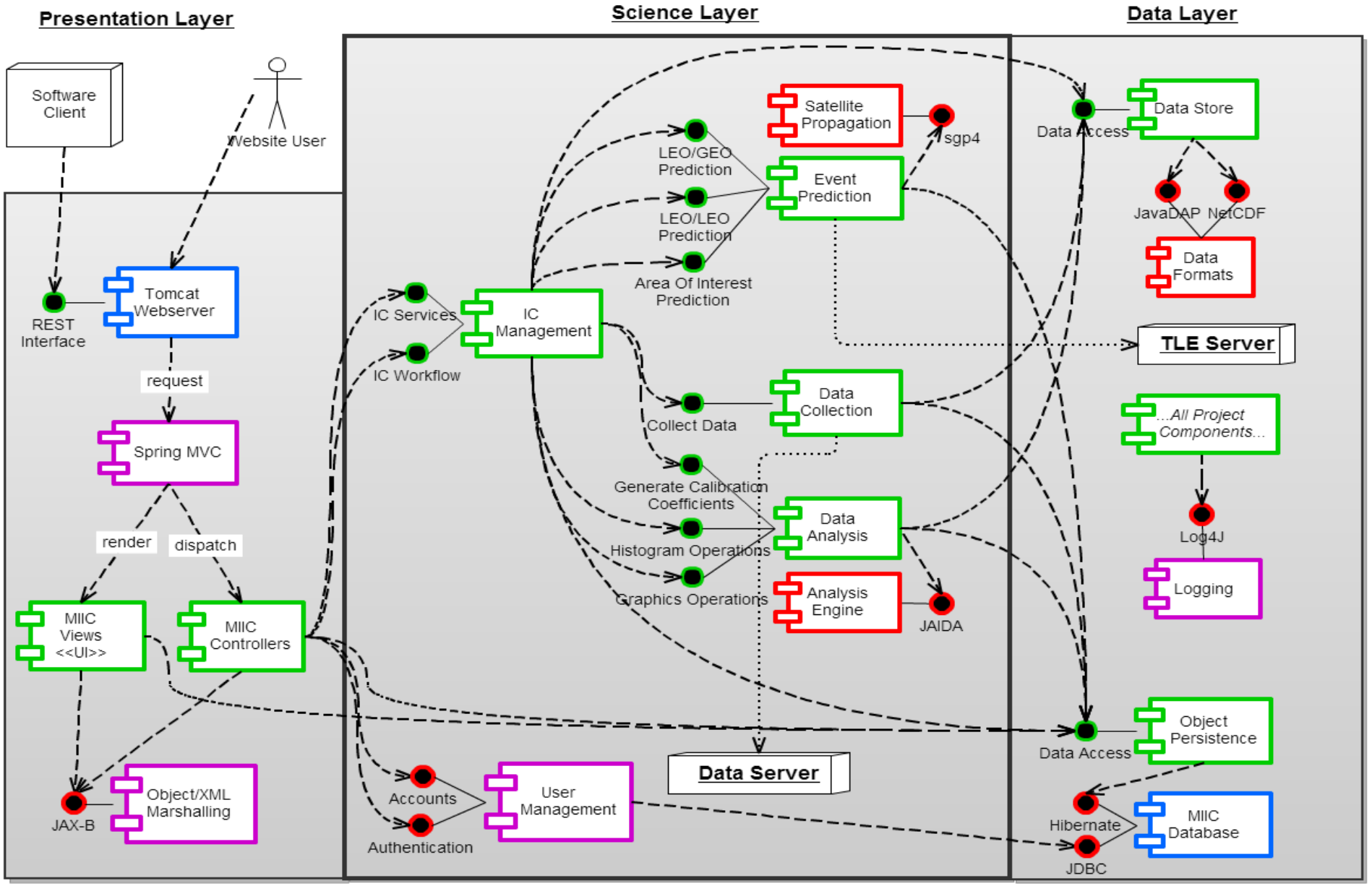
Technology Stack: Client

- HTML5 + CSS + Javascript
- jQueryUI interface components
  - Interactive, attractive, simplifies server-side
  - AJAX + JSON for dynamic data
- REST API
  - Can be written in any language, uses security
  - XML + netCDF are data transfer formats
- Google Maps (might replace with cesium)

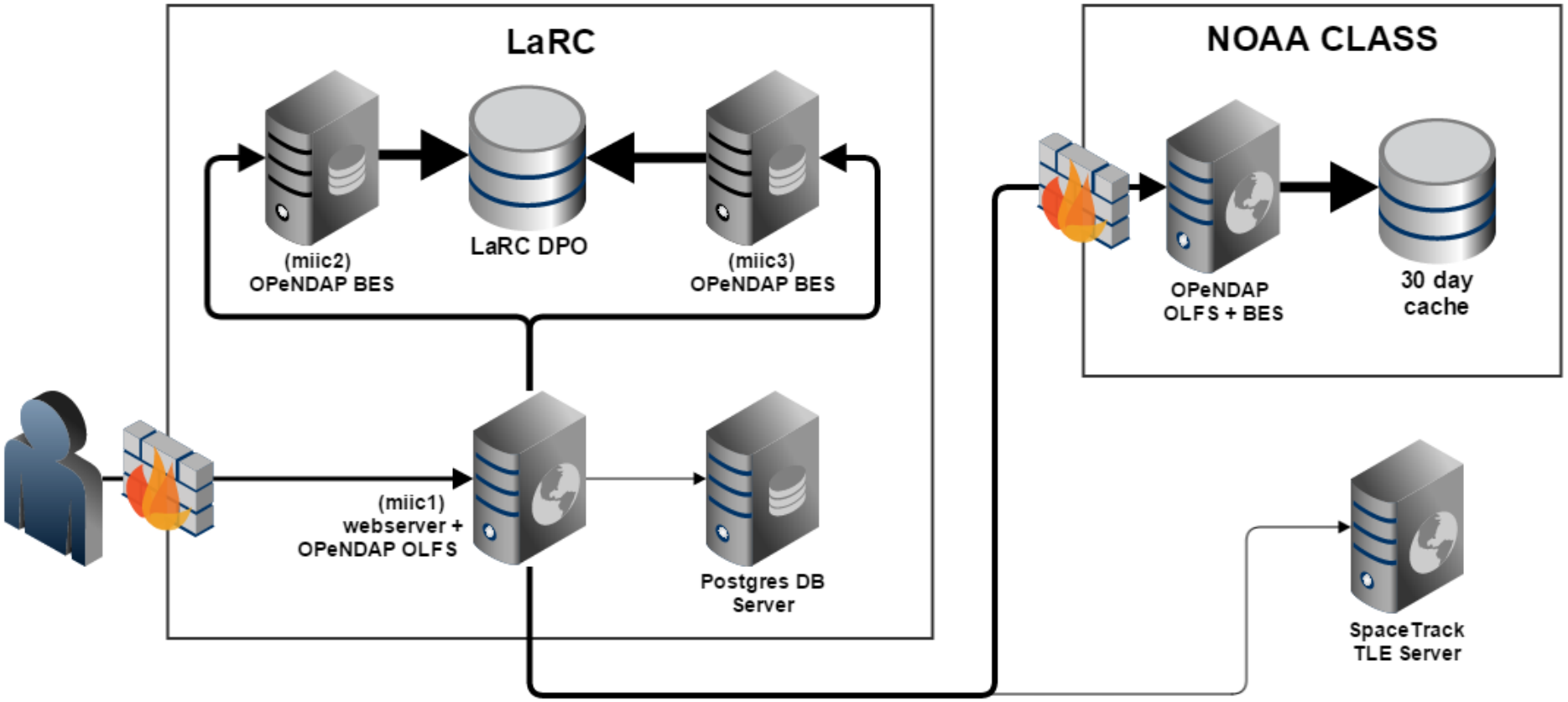
Current MIIC Stats

- Tomcat Webapp
  - ~20 second startup
- Database size
  - ~8M initial size, 17 tables
- Java code stats via Eclipse Metrics plugin
  - 14K LOC
  - 190 classes, avg. 10.5 methods per class

Application Tier Component-Based Design



MIIC Deployment Plan



MIIC Deployment Issues

- Data Volume
- Security
- Authentication
- Availability
- Scalability
- Maintenance

Data Volume

- Expected low number of users
- Very low MIIC DB usage. DB stores only metadata (data product metadata, user-owned plans and associated entities)
- No policy on how much data one user can access
- Plan data cached locally and available for pick-up when plan is completed
  - Users retrieve data zip via HTTP request
  - Need timer to discard abandoned user data
- Investigate integrating with ANGe FTP pick-up capability
  - Offload data downloading from our server
  - Offload data storage & maintenance also

Possible Data Policies

- Limit data returned by OPeNDAP query on a single file
  - Data size influenced by combination of:
    - Number of histogram bins
    - Filters – in particular, lat/lon bounding area
    - Number of data variables selected
  - Difficult to predict the effect of filters on resulting data size
  - *Stop plans if (for example) the query return data > 20% of the file size*
- Limit volume of data retrieved per user
  - Volume depends on # events, files per event, size of file query
  - *Pause plans after daily limit is exceeded, allow to resume later*
  - *Stop plans if we can predict total size exceeds limit*

Security

- OPeNDAP security currently not enabled
  - Security primarily through webserver config (i.e. Apache)
  - *May want to restrict outside access to "/opendap" URL*
- MIIC webapp uses Spring security
  - All URL access goes through spring (including REST)
    - Static resource files (images, etc.) bypass security
    - All other URLs use declarative role-based security: ROLE_USER or ROLE_ADMIN
    - Service layer verifies authenticated user vs. owned entities
- Other
  - Webapp uses "remember me" tokens which expire after 14 days

Authentication (Webapp)

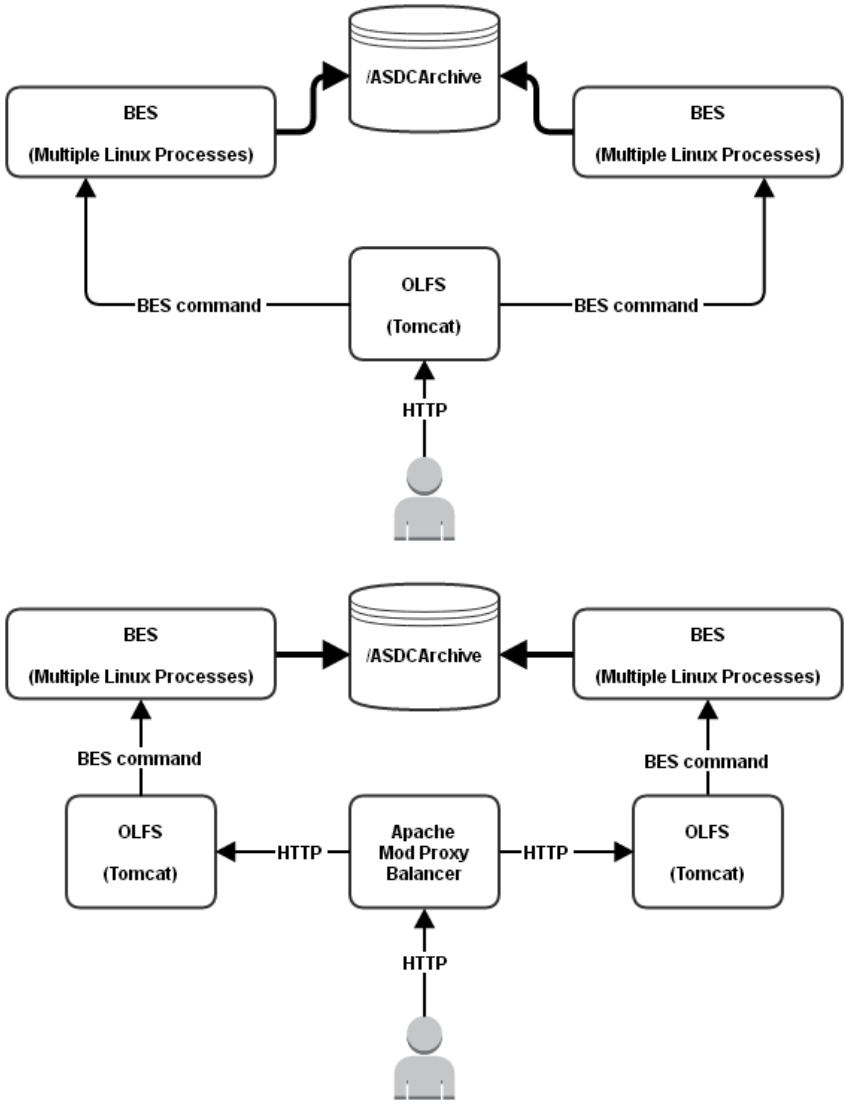
- Local database authentication (form based)
- NASA URS authentication (OAuth2) on branch
  1. Accessing an unauthorized URL redirects to URS for login
  2. User must grant access to MIIC application (one time only)
  3. URS redirects back to MIIC with authorization token
  4. MIIC requests access token from URS
  5. MIIC requests user profile from URS
- Regardless of auth scheme, MIIC stores additional user record in database (needed for ROLE, if not password)

Authentication (OPeNDAP)

- Authentication can also be added to OPeNDAP servers
  - http://docs.opendap.org/index.php/Hyrax_-_User_Identification_%28Authentication%29
  - Managed by Apache webserver plugins
  - LDAP & OAuth2 might be of interest to data center…

Availability/Scalability (OPeNDAP)

- One OPeNDAP query per core
  - 32+32 simultaneous queries
- MIIC shares the (64) open slots among all users round-robin
- OPeNDAP config options
  - Currently using single OLFS (round-robin)
    - *Doesn't handle crashed BES*
  - Apache proxy balancer should handle crashed BES



Availability/Scalability (Webapp)

- Event prediction is CPU intensive task
  - Algorithm predicts events at 1 day interval
  - Limited to # cores (@miic1, 32 simultaneous days)
  - Time to run 1 day influenced by users' event prediction options
- Scale via multiple MIIC webapps (unlikely)
  - Currently would need load balancers with sticky sessions
- Scale via terracotta cluster (more likely)
  - Distributed processing tasks (e.g. speed up large event predictions)
  - Distributed data cache (future use-cases -- processing tasks that must run on large data!)

Availability/Scalability (Other)

- SpaceTrack limits users to max queries per minute
  - Should not be a concern, we can pre-load all TLEs if necessary
- OPeNDAP data file availability issues
  - Multiple OPeNDAP servers are supported
  - No a-priori knowledge of archive structure is required (MIIC will crawl OPeNDAP servers to find files for a data collection)
  - Can achieve redundancy by serving the same file from multiple OPeNDAP servers

Maintenance/Operations

- Support for new data collections can be added in a spring XML config file, automatically uploaded to the DB at startup
  - Detailed product metadata from OPeNDAP file metadata
- Java properties files
  - Credentials (database, space-track, URS app secret key)
  - OPeNDAP server locations and some tunable settings
- Webapp ROLE_ADMIN features
  - Full access to all user plans
- Jconsole (if Tomcat configured to allow connection)
  - Clear OPeNDAP cache (necessary if location of files on server changes)
  - Change log level (e.g. enable additional debug logging)

Deployment to-do list

- ❑ Enable secure JConsole access
- ❑ Merge URS Authentication Branch?
  - ❑ Admin page to grant ROLE_USER (control who has access)
  - ❑ Test REST access w/ URS cookie
- ❑ Security
  - ❑ If using OAuth2 & secure tokens, is https required?
  - ❑ Independent audit?
  - ❑ Open up /miic URL to the web
- ❑ Define & Implement Data Policies
  - ❑ Maximum data on a per-file basis?
  - ❑ Maximum data on a per-user basis?
  - ❑ Clean-up cached data